CFM1901US
USAN 09/559,656

日 本 国 特 許 庁
PATENT OFFICE
JAPANESE GOVERNMENT

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office.

出願年月日
Date of Application: １９９９年 ４月２７日

出願番号
Application Number: 平成１１年特許願第１２０７１３号

出願人
Applicant (s): キヤノン株式会社

CERTIFIED COPY OF
PRIORITY DOCUMENT

２０００年 ５月１９日

特許庁長官
Commissioner,
Patent Office 近藤隆彦

出証番号 出証特２０００－３０３６０１６

【書類名】　特許願

【整理番号】　3960007

【提出日】　平成11年 4月27日

【あて先】　特許庁長官殿

【国際特許分類】　G06F　3/00

【発明の名称】　データ処理方法及び装置及び記憶媒体

【請求項の数】　28

【発明者】

【住所又は居所】　東京都大田区下丸子３丁目３０番２号　キヤノン株式会社内

【氏名】　榎田　幸

【発明者】

【住所又は居所】　東京都大田区下丸子３丁目３０番２号　キヤノン株式会社内

【氏名】　草間　澄

【特許出願人】

【識別番号】　000001007

【氏名又は名称】　キヤノン株式会社

【代理人】

【識別番号】　100076428

【弁理士】

【氏名又は名称】　大塚　康徳

【電話番号】　03-5276-3241

【選任した代理人】

【識別番号】　100093908

【弁理士】

【氏名又は名称】　松本　研一

【電話番号】　03-5276-3241

【選任した代理人】

【識別番号】　100101306

【弁理士】

【氏名又は名称】　丸山　幸雄

【電話番号】　03-5276-3241

【手数料の表示】

【予納台帳番号】　003458

【納付金額】　21,000円

【提出物件の目録】

【物件名】　明細書　1

【物件名】　図面　1

【物件名】　要約書　1

【包括委任状番号】　9704672

【プルーフの要否】　要

【書類名】　明細書

【発明の名称】　データ処理方法及び装置及び記憶媒体

【特許請求の範囲】

【請求項１】　ディレクトリ構造でファイルを管理するファイル管理システムにおいて各ディレクトリ毎に用意されているディレクトリデータを読み込む第１読込工程と、

前記ディレクトリデータに付与すべきメタデータを読み込む第２読込工程と、

前記第１読込工程で読み込まれたディレクトリデータの後に、前記第２読込工程で読み込まれたメタデータを接続する接続工程と、

前記接続工程によって得られたデータの全体をディレクトリデータファイルとして出力する出力工程と

を備えることを特徴とするデータ処理方法。

【請求項２】　前記第２読込工程で読み込まれたメタデータが、所定のデータ記述言語における適正な形式で記述されているか否かを判定する判定工程を更に備え、

前記接続工程は、前記判定工程で適正な形式で記述されていると判定された場合に、前記メタデータを前記バイナリデータの後に接続する

ことを特徴とする請求項１に記載のデータ処理方法。

【請求項３】　前記判定工程は、前記メタデータが前記所定のデータ記述言語としての正当性を満足するか否かを含めて判定する

ことを特徴とする請求項２に記載のデータ処理方法。

【請求項４】　ディレクトリデータファイルのデータにメタデータが登録されているか否かを判別する方法であって、

ディレクトリデータファイルを読み込む読込工程と、

前記読込工程で読み込まれたディレクトリデータファイルのデータを末尾より検査し、所定のデータ記述言語における適正な形式で記述されたデータが存在するか否かを判定することにより、該データに含まれるメタデータを判別する判別工程と

を備えることを特徴とするデータ処理方法。

【請求項５】　前記判別工程においてメタデータが判別された場合、判別されたメタデータを抽出して出力する出力工程を更に備える

ことを特徴とする請求項４に記載のデータ処理方法。

【請求項６】　前記出力工程は、前記抽出されたメタデータに基づく表示を行う

ことを特徴とする請求項５に記載のデータ処理方法。

【請求項７】　前記出力工程は、前記抽出されたメタデータを、前記所定のデータ記述言語に基づいて所定の処理を実行するためのツールに提供する

ことを特徴とする請求項４に記載のデータ処理方法。

【請求項８】　前記判別工程は、

前記所定のデータ記述言語によって規定された末尾文字列が前記データの末尾に存在するか否かをチェックするチェック工程と、

該末尾文字列が存在する場合に前記所定のデータ記述言語に規定された先頭文字列を該データの先頭へ向かって検索する検索工程とを備え、

前記末尾文字列と前記先頭文字列によって挟まれたデータが存在する場合に、このデータをメタデータと判別する

ことを特徴とする請求項４に記載のデータ処理方法。

【請求項９】　前記判別工程は、前記末尾文字列と前記先頭文字列によって挟まれたデータが、前記所定のデータ記述言語における適正な形式を有するか否かを検査する検査工程を更に備える

ことを特徴とする請求項８に記載のデータ処理方法。

【請求項１０】　前記検査工程は、前記所定のデータ記述言語としての正当性を満足するかの検査もあわせて行う

ことを特徴とする請求項９に記載のデータ処理方法。

【請求項１１】　前記所定のデータ記述言語がＸＭＬである

ことを特徴とする請求項１乃至９のいずれかに記載のデータ処理方法。

【請求項１２】　前記所定のデータ記述言語がＳＧＭＬである

ことを特徴とする請求項１乃至９のいずれかに記載のデータ処理方法。

【請求項１３】　前記所定のデータ記述言語がＨＴＭＬである

ことを特徴とする請求項１乃至９のいずれかに記載のデータ処理方法。

【請求項１４】　ファイル管理システムによるディレクトリデータの更新方法であって、

更新が必要なディレクトリデータファイルを読み込む読込工程と、

前記読込工程で読み込まれたディレクトリデータファイルのデータを末尾より検査し、所定のデータ記述言語における適正な形式で記述されたデータが存在するか否かを判定することにより、該データに含まれるメタデータを判別する判別工程と、

前記判別工程でメタデータが存在すると判定された場合、当該判別されたメタデータを分離する分離工程と、

前記分離工程によってメタデータが分離されたデータに対して必要な更新処理を施す更新工程と、

前記更新工程によって更新されたデータの後に、前記分離工程で分離されたメタデータを接続する接続工程と

を備えることを特徴とするデータ処理方法。

【請求項１５】　ディレクトリ構造でファイルを管理するファイル管理システムにおいて各ディレクトリ毎に用意されているディレクトリデータを読み込む第１読込手段と、

前記ディレクトリデータに付与すべきメタデータを読み込む第２読込手段と、

前記第１読込手段で読み込まれたディレクトリデータの後に、前記第２読込手段で読み込まれたメタデータを接続する接続手段と、

前記接続手段によって得られたデータの全体をディレクトリデータファイルとして出力する出力手段と

を備えることを特徴とするデータ処理装置。

【請求項１６】　前記第２読込手段で読み込まれたメタデータが、所定のデータ記述言語における適正な形式で記述されているか否かを判定する判定手段を更に備え、

前記接続手段は、前記判定手段で適正な形式で記述されていると判定された場合に、前記メタデータを前記バイナリデータの後に接続する

ことを特徴とする請求項１５に記載のデータ処理装置。

【請求項１７】　前記判定手段は、前記メタデータが前記所定のデータ記述言語としての正当性を満足するか否かを含めて判定する

ことを特徴とする請求項１６に記載のデータ処理装置。

【請求項１８】　ディレクトリデータファイルのデータにメタデータが登録されているか否かを判別する方法であって、

ディレクトリデータファイルを読み込む読込手段と、

前記読込手段で読み込まれたディレクトリデータファイルのデータを末尾より検査し、所定のデータ記述言語における適正な形式で記述されたデータが存在するか否かを判定することにより、該データに含まれるメタデータを判別する判別手段と

を備えることを特徴とするデータ処理装置。

【請求項１９】　前記判別手段においてメタデータが判別された場合、判別されたメタデータを抽出して出力する出力手段を更に備える

ことを特徴とする請求項１８に記載のデータ処理装置。

【請求項２０】　前記出力手段は、前記抽出されたメタデータに基づく表示を行う

ことを特徴とする請求項１９に記載のデータ処理装置。

【請求項２１】　前記出力手段は、前記抽出されたメタデータを、前記所定のデータ記述言語に基づいて所定の処理を実行するためのツールに提供する

ことを特徴とする請求項２０に記載のデータ処理装置。

【請求項２２】　前記判別手段は、

前記所定のデータ記述言語によって規定された末尾文字列が前記データの末尾に存在するか否かをチェックするチェック手段と、

該末尾文字列が存在する場合に前記所定のデータ記述言語に規定された先頭文字列を該データの先頭へ向かって検索する検索手段とを備え、

前記末尾文字列と前記先頭文字列によって挟まれたデータが存在する場合に、このデータをメタデータと判別する

ことを特徴とする請求項１９に記載のデータ処理装置。

【請求項２３】　前記判別手段は、前記末尾文字列と前記先頭文字列によって挟まれたデータが、前記所定のデータ記述言語における適正な形式を有するか否かを検査する検査手段を更に備える

ことを特徴とする請求項２２に記載のデータ処理装置。

【請求項２４】　前記検査手段は、前記所定のデータ記述言語としての正当性を満足するかの検査もあわせて行う

ことを特徴とする請求項２３に記載のデータ処理装置。

【請求項２５】　ファイル管理システムによるディレクトリデータの更新方法であって、

更新が必要なディレクトリデータファイルを読み込む読込手段と、

前記読込手段で読み込まれたディレクトリデータファイルのデータを末尾より検査し、所定のデータ記述言語における適正な形式で記述されたデータが存在するか否かを判定することにより、該データに含まれるメタデータを判別する判別手段と、

前記判別手段でメタデータが存在すると判定された場合、当該判別されたメタデータを分離する分離手段と、

前記分離手段によってメタデータが分離されたデータに対して必要な更新処理を施す更新手段と、

前記更新手段によって更新されたデータの後に、前記分離手段で分離されたメタデータを接続する接続手段と

を備えることを特徴とするデータ処理装置。

【請求項２６】　ディレクトリデータにメタデータを登録する処理のためのコンピューターログラムを格納する記憶媒体であって、該コンピュータプログラムが、

ディレクトリ構造でファイルを管理するファイル管理システムにおいて各ディレクトリ毎に用意されているディレクトリデータを読み込む第１読込工程のコードと、

前記ディレクトリデータに付与すべきメタデータを読み込む第２読込工程のコードと、

前記第１読込工程で読み込まれたディレクトリデータの後に、前記第２読込工程で読み込まれたメタデータを接続する接続工程のコードと、

前記接続工程によって得られたデータの全体をディレクトリデータファイルとして出力する出力工程のコードとを備えることを特徴とする記憶媒体。

【請求項２７】　ディレクトリデータファイルのデータにメタデータが登録されているか否かを判別する処理のためのコンピュータプログラムを格納する記憶媒体であって、該コンピュータプログラムが、

ディレクトリデータファイルを読み込む読込工程のコードと、

前記読込工程で読み込まれたディレクトリデータファイルのデータを末尾より検査し、所定のデータ記述言語における適正な形式で記述されたデータが存在するか否かを判定することにより、該データに含まれるメタデータを判別する判別工程のコードとを備えることを特徴とする記憶媒体。

【請求項２８】　ファイル管理システムによるディレクトリデータの更新処理のためのコンピュータプログラムを格納する記憶媒体であって、

更新が必要なディレクトリデータファイルを読み込む読込工程のコードと、

前記読込工程で読み込まれたディレクトリデータファイルのデータを末尾より検査し、所定のデータ記述言語における適正な形式で記述されたデータが存在するか否かを判定することにより、該データに含まれるメタデータを判別する判別工程のコードと、

前記判別工程でメタデータが存在すると判定された場合、当該判別されたメタデータを分離する分離工程のコードと、

前記分離工程によってメタデータが分離されたデータに対して必要な更新処理を施す更新工程のコードと、

前記更新工程によって更新されたデータの後に、前記分離工程で分離されたメタデータを接続する接続工程のコードとを備えることを特徴とするコンピュータプログラム。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】

本発明は、データにメタデータを登録、判別することを可能とするデータ処理方法および装置に関する。

【０００２】

【従来の技術】

メタデータ（meta-data）とは、「データに関するデータ」であり、画像データや音声データ等のバイナリデータを説明するデータとして用いられている。しかし、バイナリデータとこれに対応するメタデータが別々のファイルで存在した場合、ファイルの移動やコピーの際に、ユーザはバイナリデータとメタデータとを同時に管理しなければならず、非常にわずらわしいことになる。

【０００３】

そこで一般に、バイナリデータとメタデータの管理を容易にするために、バイナリデータとメタデータを記述する様々な方法が提案されてきた。この種の従来技術は、新しいバイナリフォーマットを規定する方法と、データベースで管理する方法の２つに分けることができる。

【０００４】

まず、新しいバイナリフォーマットを規定する方法の一例をあげると、画像フォーマットではTiff、Exif、Flashpixなどがある。図９は、バイナリデータにメタデータを埋め込んだフォーマットの概観を示す図である。バイナリデータとしては、例えば画像データが挙げられる。図９に示されるように、画像のヘッダ部分にメタデータを記述する枠組みを設け、そこにユーザがメタデータを記述するというのが一般的な方法である。このようにメタデータを記述することにより、データの検索・分類が容易になる。また、バイナリデータ内にメタデータを含むようになるので、１つのファイルで管理でき、ファイルの管理は比較的容易になる。

【０００５】

次に、バイナリデータとメタデータをデータベースで管理する方法を説明する。図１０はバイナリデータとメタデータをデータベースで管理する方法を概念的に示した図である。図１０に示されるような、別々のファイルで存在するバイナリデータとメタデータをデータベース等を用いて管理するという方法も広く行わ

れているものである。この場合は既存のバイナリデータが、既存のアプリケーションでそのまま使えるという利点がある。

【０００６】

また、ファイル管理においてディレクトリ構造が用いられ得ることはよく知られている。従って、例えば大量のバイナリ・データファイルを管理する場合は、通常１つのディレクトリの下に全てのバイナリ・データファイルを置くようなことはせず、あるまとまった属性や特性を持つバイナリ・データファイル群毎にサブディレクトリを作成し、その下に複数のバイナリ・データファイルを置くことが多い。この場合、１つのサブディレクトリの下に置かれている各バイナリ・データファイルのメタデータには共通の項目があることになる。しかしながら、現状では、各バイナリ・データファイルの各々が独立に、メタデータを上記２つの方式のいずれかの方法により管理していた。

【０００７】

【発明が解決しようとする課題】

しかしながら、上述したようなメタデータを記述する新フォーマットを規定する方法とデータベースを用いてメタデータを管理する方法のそれぞれに問題がある。

【０００８】

まず、メタデータを記述する新フォーマットを規定した場合には、既存のバイナリデータを当該新フォーマットに変換し、なおかつその新フォーマット内にメタデータを記述しなければならない。更に、その新フォーマット内のメタデータを用いて検索するためには、当該新フォーマット対応のアプリケーションが必要となる。すなわち、メタデータを記述したり利用したりするために、非常に多くのステップと専用の環境が必要になるという問題がある。また、このような新フォーマットのバイナリデータを処理する（例えば画像データであれば画像の再生）ためには、当該フォーマットに対応したアプリケーションが必要であり、既存のアプリケーションでは対応できなくなる。

【０００９】

そのうえ、メタデータの記述方法も新フォーマットにおいて独自に決められた

ものであり、新フォーマット内のメタデータを利用するアプリケーションを作成するためには、新規にメタデータの検索ルーチンをつくらなければならないという問題もある。さらに、新しい枠組みのメタデータを記述するにはフォーマットの規定を変更しなければならないという問題点もあった。

【００１０】

一方、データベースを用いてバイナリデータとメタデータを同時に管理する場合、データベースソフトが無ければメタデータの登録も利用もできないという問題があった。また、登録したメタデータを表示するためにも専用のソフトウェアが必要である。更に、バイナリデータをデータベース外に持っていくと、メタデータは付加されず、メタデータのないバイナリデータとなってしまうという問題点もあった。

【００１１】

更に、ディレクトリ構造でバイナリ・データファイルを管理する場合においても、１つのディレクトリの下に置かれているバイナリ・データファイルのメタデータは、各バイナリ・データファイル毎に存在し、それぞれ独立して管理される。このため、複数のバイナリ・データファイルに存在する共通項目についても各バイナリ・データファイル毎独立して管理することになり、管理するメタデータのデータ量が多くなってしまうという問題がある。また、その共通項目を変更する場合には、全てのバイナリ・データファイルのメタデータに対して逐一変更を行わなければならないという欠点もある。

【００１２】

本発明はメタデータの記述・検索に関する上記の問題点に鑑みてなされたものであり、既存のアプリケーションに影響を与えずに、ディレクトリデータにメタデータを登録可能とすることを目的とする。

【００１３】

また、本発明の他の目的は、メタデータの記述に一般的なデータ記述言語を用いることにより、データ記述言語用の既存のツールを利用することを可能とし、対応アプリケーションの開発を容易にすることにある。

【００１４】

また、本発明の他の目的は、メタデータが記述されたディレクトリデータからメタデータを抽出し、例えば検索、参照、変更等の処理に供することを可能とすることにある。

【００１５】

また、本発明の他の目的は、ディレクトリに対してメタデータを登録することにより、メタデータの管理容易化、データ量の削減を図ることにある。

【００１６】

更に、本発明の他の目的は、ある属性あるいは特性などによりグループ化されたディレクトリにより管理されているデータファイルに対して簡便にメタデータを付属可能とすることにある。

【００１７】

【課題を解決するための手段】

上記の目的を達成するための本発明のデータ処理方法はたとえば以下の工程を備える。すなわち、

ディレクトリ構造でファイルを管理するファイル管理システムにおいて各ディレクトリ毎に用意されているディレクトリデータを読み込む第１読込工程と、

前記ディレクトリデータに付与すべきメタデータを読み込む第２読込工程と、

前記第１読込工程で読み込まれたディレクトリデータの後に、前記第２読込工程で読み込まれたメタデータを接続する接続工程と、

前記接続工程によって得られたデータの全体をディレクトリデータファイルとして出力する出力工程とを備える。

【００１８】

また、上記の目的を達成するための本発明の他の形態によるデータ処理方法は、例えば以下の工程を備える。すなわち、

ディレクトリデータファイルのデータにメタデータが登録されているか否かを判別する方法であって、

ディレクトリデータファイルを読み込む読込工程と、

前記読込工程で読み込まれたディレクトリデータファイルのデータを末尾より検査し、所定のデータ記述言語における適正な形式で記述されたデータが存在す

るか否かを判定することにより、該データに含まれるメタデータを判別する判別工程とを備える。

【００１９】

更に、上記の目的を達成するための本発明の他の形態によるデータ処理方法は、例えば以下の工程を備える。すなわち、

ファイル管理システムによるディレクトリデータの更新方法であって、

更新が必要なディレクトリデータファイルを読み込む読込工程と、

前記読込工程で読み込まれたディレクトリデータファイルのデータを末尾より検査し、所定のデータ記述言語における適正な形式で記述されたデータが存在するか否かを判定することにより、該データに含まれるメタデータを判別する判別工程と、

前記判別工程でメタデータが存在すると判定された場合、当該判別されたメタデータを分離する分離工程と、

前記分離工程によってメタデータが分離されたデータに対して必要な更新処理を施す更新工程と、

前記更新工程によって更新されたデータの後に、前記分離工程で分離されたメタデータを接続する接続工程とを備える。

【００２０】

また、本発明の他の態様によれば、上記のデータ処理方法を実現するデータ処理装置が提供される。さらに本発明の他の態様によれば、上記データ処理方法をコンピュータに実現させるためのコンピュータプログラムを格納した記憶媒体が提供される。

【００２１】

【発明の実施の形態】

以下、添付の図面を参照して本発明の好適な実施形態を説明する。

【００２２】

＜第１の実施形態＞

図１は第１の実施形態によるデータ処理装置の構成を示すブロック図である。図１において、１００は読込部であり、スキャナ装置などを用いて画像を読み込

む。１０１は入力部であり、ユーザからの指示やデータを入力するもので、キーボードやポインティング装置を含む。１０２は蓄積部であり、バイナリデータやメタデータをディレクトリ構造で蓄積する。蓄積部１０２としては、ハードディスクを用いるのが一般的である。１０３は表示部であり、蓄積部１０２に蓄積されたバイナリデータを表示したり、読込部１００で読み込まれた画像データを表示する。表示部１０３としては、ＣＲＴや液晶表示装置が一般的である。

【００２３】

１０４はＣＰＵであり、上述した各構成の処理のすべてに関わり、ＲＯＭ１０５とＲＡＭ１０６はその処理に必要なプログラム、データ、或いは作業領域をＣＰＵ１０４に提供する。なお、図２のフローチャートを参照して後述する本実施形態の処理手順を実現するための制御プログラムもＲＯＭ１０５に格納されているものとする。もちろん、蓄積部１０２にその制御プログラムを格納しておき、ＣＰＵ１０４による実行に応じてその制御プログラムがＲＡＭ１０６上へロードされるような構成であってもよい。

【００２４】

なお、第１の実施形態のデータ処理装置には上記以外にも、種々の構成要素が設けられているが、本発明の主眼ではないので、その説明については省略する。

【００２５】

つぎに、以上のように構成されたデータ処理装置において、メタデータをバイナリデータに登録する処理について説明する。図２は、第１の実施形態によるメタデータの登録処理を説明するフローチャートである。

【００２６】

図２において、まず、ステップＳ３０１で、ユーザによって指定されたディレクトリのディレクトリデータをメモリ（ＲＡＭ１０６）上に読み込む。これは例えば所望のディレクトリ名をキーボードから入力したり、ポインティング装置（例えばマウス）によって当該ディレクトリを指示することによりなされる。次にステップＳ３０２において、ユーザによって指定された、メタデータが記述されているＸＭＬファイルをメモリ（ＲＡＭ１０６）上に読み込む。このＸＭＬファイルの指定も、キーボードからファイル名を入力したり、ポインティング装置（

例えばマウス）で対応するアイコンを指示する等によって行われる。

【００２７】

次にステップＳ３０３で、メタデータを記述したＸＭＬファイルが適正形式のＸＭＬデータであるかを調べる。この適性形式の判定では、ＸＭＬファイルの記述フォーマットを満足しているか（例えば、タグの左右の括弧が正しく対をなしているか、タグ付けの形式が正しいかどうか等）がチェックされる。なお、適性形式のＸＭＬデータであるか否かの判定は、正当なＸＭＬデータであるか否かを含めたチェックであってもよい。ここで、正当なＸＭＬデータか否かの判定は、例えば、ＸＭＬデータがＤＴＤ（Document Type Definition）等のスキーマに従って記述されているかどうか等のチェックを行うことでなされる。

【００２８】

ステップＳ３０３において適正形式のＸＭＬデータでないと判定された場合にはステップＳ３０５に進む。ステップＳ３０５では、ＸＭＬデータにエラーがある旨を表示部１０３に表示し、本処理を終了する。

【００２９】

一方、ステップＳ３０３においてＸＭＬファイルが適正形式のＸＭＬデータであると判定された場合には、処理はステップＳ３０４に進む。ステップＳ３０４では、ステップＳ３０１でメモリ上に読み込まれたディレクトリデータの後ろに当該メタデータを接続することにより、メタデータの登録を行う。その後、ステップＳ３０６において、メタデータを登録したディレクトリデータを出力し、処理を終了する。なお、ステップＳ３０６におけるデータ出力により、図３に示されるデータ構造を有するメタデータ付ディレクトリデータが１つのディレクトリデータファイルとして蓄積部１０２に格納されることになる。

【００３０】

図３は本実施形態によるディレクトリデータへのメタデータの登録状態を説明する図である。図３に示されるように、ディレクトリデータの最後に（本例では、ディレクトリデータの終端を示す識別子＜ＥＯＦ＞の後に）、ＸＭＬデータで記述されたメタデータを接続する。このようにすることにより、他のアプリケーションには影響を与えずに、メタデータをディレクトリデータに登録することが

できる。すなわち、一般のアプリケーションはディレクトリデータを参照する場合に、ディレクトリデータの先頭から終端識別子までのデータを用いるので、接続されたメタデータは何等影響を及ぼさないのである。

【００３１】

さらに、メタデータはＸＭＬで記述されているため、このＸＭＬデータ部分を抽出しておくことにより、ＸＭＬデータを理解するツールがあれば、メタデータの追加・変更・参照が可能であり、非常に汎用性に優れている。なお、ディレクトリデータからＸＭＬデータ部分を抽出する処理については第２の実施形態で詳しく説明する。

【００３２】

以上説明したように、第一の実施形態によれば、メタデータをＸＭＬで記述し、ディレクトリデータの最後に接続することにより、既存のアプリケーションに影響を及ぼさずに、既存のディレクトリデータにメタデータを登録することができる。

【００３３】

＜第２の実施形態＞

第１の実施形態においてディレクトリデータにメタデータを登録して、ディレクトリデータファイルとする方法を説明した。第２の実施形態では、ディレクトリデータファイルにメタデータが登録されているかどうかを判別し、登録されている場合にはそのメタデータを抽出する処理について説明する。なお、第２の実施形態におけるデータ処理装置の構成は第１の実施形態（図１）と同様であるのでここでは説明を省略する。

【００３４】

以下、指定されたディレクトリのディレクトリデータファイルに第１の実施形態で説明した如きメタデータが登録されているか否かの判定と、登録されたメタデータを抽出する動作について説明する。図４は第２の実施形態による登録されたメタデータの判別及び抽出手順を示すフローチャートである。なお、本実施形態では、抽出されたメタデータを表示部１０３に表示するが、出力の形態はこれに限らない。例えば、抽出したメタデータを編集や検索等の処理に提供するよう

に構成してもよいことは当業者には明らかであろう。

【００３５】

図４によれば、まず、ステップＳ５０１で、ユーザの指示により、メタデータが登録されているかを判別したいディレクトリデータファイル、即ち処理対象データを指定する。ステップＳ５０１における、処理対象データの指定は、キーボードから当該バイナリデータのファイル名を入力したり、対応するアイコンをポインティング装置（マウス）で指示することにより行われる。

【００３６】

次にステップＳ５０２において、指定されたディレクトリデータファイルのデータにＸＭＬで記述されたメタデータが登録されているかどうかを判別する。以下、ステップＳ５０２における判別処理の詳細について図５のフローチャートと、図６の概略図にしたがって説明する。図５は第２の実施形態によるメタデータの判別処理の詳細を説明するフローチャートである。また、図６はメタデータとしてＸＭＬデータが登録されたディレクトリデータのデータ構成例を示す図である。

【００３７】

第１の実施形態で説明したように、メタデータとしてのＸＭＬデータが登録されているディレクトリデータファイル（処理対象データ）のデータ構成は図６のようになっている。したがって、メタデータの有無の判別は以下のように行われる。

【００３８】

図５に示されるように、まず、ステップＳ６０１で、上記ステップＳ５０１で指定されたディレクトリデータファイルのデータ全体（処理対象データの全体）をメモリ（ＲＡＭ１０６）上に読み込む。なお、第１の実施形態のステップＳ３０６によって出力されたデータは一つのディレクトリデータファイルとして管理されるので、ファイル管理システムによってこのデータの全体を読出すことが可能である。

【００３９】

次にステップＳ６０２において、ステップＳ６０１で読み込んだデータの最後

に“＜/PhotoXML＞”という文字列があるか調べる。存在しなかった場合はステップＳ６０５に進む。

【００４０】

一方、読み込んだ処理対象データの最後に、“＜/PhotoXML＞”という文字列が存在した場合はステップＳ６０３にすすむ。ステップＳ６０３では“＜/PhotoXML＞”という文字列の前に“＜PhotoXML＞”という文字列が存在するかどうかを調べる。

【００４１】

ステップＳ６０３において“＜PhotoXML＞”という文字列の存在が確認された場合は、ステップＳ６０４にすすむ。ステップＳ６０４においては、メタデータが登録されているものと結論づけ、本処理を終了する。一方、ステップＳ６０３において“＜PhotoXML＞”という文字列の存在が確認されなかった場合には、処理はステップＳ６０５に進む。ステップＳ６０５においては、メタデータは登録されていないものと結論づける。すなわち、ステップＳ６０２で、当該バイナリデータの最後に文字列“＜/PhotoXML＞”が存在しない場合、ステップＳ６０３で文字列“＜PhotoXML＞”が存在しない場合には、処理はステップＳ６０５に進み、当該処理対象データにメタデータは登録されていないものと結論づける。

【００４２】

なお、ステップＳ６０３において、“＜PhotoXML＞”という文字列の存在が確認された後に、さらにそれらの文字列で囲まれたデータが、ＸＭＬの適正形式で記述されているかを確認するようにしてもよい。更に、この場合、当該データがＸＭＬの正当なデータであるか否かの判定を含めて行うようにしてもよい。適性形式か否かの判定、正当なデータか否かの判定は、第１の実施形態（ステップＳ３０３）で説明したとおりである。

【００４３】

次に、図４のフローチャートにもどる。図５のフローチャートで示される処理によってメタデータが登録されていると結論づけられた場合には、処理はステップＳ５０３に進む。ステップＳ５０３では、文字列“＜PhotoXML＞”と“＜/PhotoXML＞”で囲まれた部分のＸＭＬデータに基づいて登録されているメタデータ

の内容を表示し、処理を終了する。一方、ステップＳ５０２でメタデータが登録されていないと判定された場合にはそのまま処理を終了する。

【００４４】

以上説明したように、第２の実施形態によれば、メタデータを登録したディレクトリデータと、通常のバイナリデータとを、ＸＭＬの記述規則によって判別し、メタデータが登録されているディレクトリデータの場合には、そのメタデータを表示することができる。

【００４５】

すなわち、第２の実施形態によれば、メタデータが登録されたディレクトリデータとメタデータが登録されていないディレクトリデータとを判別するとともに、登録されたメタデータを抽出することが可能となる。従って、メタデータとして既存のデータ記述言語を用いれば、メタデータの参照や編集、検索に際して、当該データ記述言語用の既存のツールをそのまま用いることができ、開発に関する手間も省くことができる。

【００４６】

次に図７を用いて具体的にディレクトリ毎に格納されているバイナリ・データファイルのメタデータを格納する方法について説明する。説明を簡単にするため、ルートディレクトリDirectory 1の下に、Directory 2, Directory 3, Directory 4の３つのサブディレクトリがあり、各サブディレクトリの下に図７に示すようにバイナリ・データファイルが置かれているものとする。

【００４７】

このような場合において、各ディレクトリのディレクトリデータにメタデータを登録すれば、Directory 2の下に置かれているBinary Data File 2-1からBinary Data File 2-3には同一のメタデータが、Directory 3の下のBinary Data File 3-1 とBinary Data File 3-2にも同一のメタデータが、同様にDirectory 4の下に置かれているファイルにも同一のメタデータが夫々保存されることになる。

【００４８】

メタデータの簡単な例を示すと、例えばディジタルカメラで撮影した場合などでは、Directory 2は、３月１０日に撮影し、Directory 3のバイナリデータは３

月１１日に撮影したといった情報である。あるいは、「風景」や「人物」等の属性でディレクトリを分けることも考えられる。いずれにせよ、サブディレクトリを作成する時のデータファイルの分類の仕方によりメタデータに格納する項目や内容を決めることができる。

【００４９】

上記の場合、各バイナリ・データファイル毎にメタデータを保存するのではなく、各サブディレクトリのディレクトリデータにメタデータが格納される。従ってメタデータを読み出したいアプリケーションは、そのファイルが属しているディレクトリに付随するメタデータを読み出し、そのバイナリ・データファイルのメタデータとして処理するように動作する。

【００５０】

例えば、メタデータに格納されているある項目でバイナリ・データファイルを検索する場合は、本実施形態によれば、ディレクトリデータに付随して格納されているメタデータがディレクトリの下に置かれているバイナリ・データファイル全てに対して有効なため、ディレクトリのみを検索すればよく、高速に検索処理を行うことが可能となる。

【００５１】

なお、ディレクトリに格納されるデータをバイナリ・データとして説明したが、いかなる種類のデータであってもよいことは明らかである。

【００５２】

＜ファイルシステムによるディレクトリデータの更新について＞

ところで、ディレクトリに対してファイルの追加や削除を行った場合、ファイルシステムはこの処理に応じてディレクトリデータを更新する。しかしながら、メタデータが接続された状態のままで更新を行うと、接続されているメタデータが破損してしまう。従って、上記第１の実施形態によってメタデータが付加されたディレクトリデータを扱うために、ファイルシステムに若干の変更が必要となる。以下、ファイル管理システムによるディレクトリデータの更新処理について説明する。

【００５３】

図８は、本実施形態のファイル管理システムによるディレクトリデータの更新処理を説明するフローチャートである。ディレクトリ内のファイルが更新、追加、削除された場合は、図８に示す処理によってディレクトリデータの更新が行われる。

【００５４】

まず、ステップＳ７０１において、当該ディレクトリのディレクトリデータファイルが読み出される。そして、ステップＳ７０２において、当該ディレクトリデータにメタデータが接続されているか否かを判定する。この判定は、上述の図５に示される手順により行うことができる。そして、メタデータが登録されていると判定された場合は、ステップＳ７０３へ進む。

【００５５】

ステップＳ７０３では、存在が検出されたメタデータを分離し、別の記憶エリアに待避させておく。そして、ステップＳ７０４では、メタデータが分離された後のディレクトリデータに対して、従前と同様の方法で編集を施す。そして、ディレクトリデータの編集を終えたならば、ステップＳ７０５において、先のステップＳ７０３で分離し待避させておいたメタデータを、更新後のディレクトリデータの後に再度接続する。一方、ステップＳ７０２においてメタデータが登録されていなければ、ステップＳ７０６へ進み、ディレクトリデータを更新する。

【００５６】

以上の処理において、ステップＳ７０４とステップＳ７０６におけるディレクトリデータの更新処理は、従前のファイル管理システムにおいて実行される処理と同一のものでよい。従って、上述のファイル管理システムの改造は大した作業ではなく、僅かな変更によって本実施形態のディレクトリファイルを取り扱い可能となることが当業者には理解されよう。

【００５７】

なお、上記各実施形態では、メタデータとしてＸＭＬデータを用いたがこれに限られるものではない。例えば、ＳＧＭＬやＨＴＭＬ等のデータ記述言語であってもよい。また、各ディレクトリに保持されるファイルのデータとしては、静止画像データ、動画像データ、音声データ等のバイナリデータや、他のいかなる形

態のデータであったもよい。

【００５８】

なお、本発明は、複数の機器（例えばホストコンピュータ，インタフェイス機器，リーダ，プリンタなど）から構成されるシステムに適用しても、一つの機器からなる装置（例えば、複写機，ファクシミリ装置など）に適用してもよい。

【００５９】

また、本発明の目的は、前述した実施形態の機能を実現するソフトウェアのプログラムコードを記録した記憶媒体を、システムあるいは装置に供給し、そのシステムあるいは装置のコンピュータ（またはＣＰＵやＭＰＵ）が記憶媒体に格納されたプログラムコードを読出し実行することによっても、達成されることは言うまでもない。

【００６０】

この場合、記憶媒体から読出されたプログラムコード自体が前述した実施形態の機能を実現することになり、そのプログラムコードを記憶した記憶媒体は本発明を構成することになる。

【００６１】

プログラムコードを供給するための記憶媒体としては、例えば、フロッピディスク，ハードディスク，光ディスク，光磁気ディスク，ＣＤ－ＲＯＭ，ＣＤ－Ｒ，磁気テープ，不揮発性のメモリカード，ＲＯＭなどを用いることができる。

【００６２】

また、コンピュータが読出したプログラムコードを実行することにより、前述した実施形態の機能が実現されるだけでなく、そのプログラムコードの指示に基づき、コンピュータ上で稼働しているＯＳ（オペレーティングシステム）などが実際の処理の一部または全部を行い、その処理によって前述した実施形態の機能が実現される場合も含まれることは言うまでもない。

【００６３】

さらに、記憶媒体から読出されたプログラムコードが、コンピュータに挿入された機能拡張ボードやコンピュータに接続された機能拡張ユニットに備わるメモリに書込まれた後、そのプログラムコードの指示に基づき、その機能拡張ボード

や機能拡張ユニットに備わるＣＰＵなどが実際の処理の一部または全部を行い、その処理によって前述した実施形態の機能が実現される場合も含まれることは言うまでもない。

【００６４】

【発明の効果】

以上説明したように、本発明によれば、メタデータをディレクトリデータに登録するのでディレクトリ単位でメタデータを登録、管理できる。

また、本発明によれば、既存のディレクトリデータの最後にメタデータを接続することにより、既存のアプリケーションに影響を与えずに、ディレクトリデータにメタデータを登録することが可能となる。

更に、本発明によれば、メタデータが記述されたディレクトリデータからメタデータを抽出し、例えば検索、参照、変更等の処理に供することが可能となる。従って、メタデータの記述に一般的なデータ記述言語を用いることにより、データ記述言語用の既存のツールを利用することが可能となり、対応アプリケーションの開発が容易である。

また、本発明によれば、ディレクトリに対してメタデータを登録することにより、メタデータの管理容易化、データ量の削減が図られる。このため、例えば、メタデータ内のある項目でバイナリデータを検索する場合には、ディレクトリのみ検索を行えばよく、高速に検索することができる。また、メタデータ内の項目に格納されている値を変更する場合には、各バイナリデータファイル毎行う必要が無く、ディレクトリデータに付属しているメタデータを変更するだけで、そのディレクトリで管理されているバイナリデータ全てに対して変更したことになり、高速に行える。また、ディレクトリで管理されている複数のバイナリデータに対して共通なメタデータを一つのメタデータで管理することができ、データ量を削減することができる。

【図面の簡単な説明】

【図１】

第１の実施形態によるデータ処理装置の構成を示すブロック図である。

【図２】

第１の実施形態によるメタデータの登録処理を説明するフローチャートである。

【図３】

実施形態によるディレクトリデータへのメタデータの登録状態を説明する図である。

【図４】

第２の実施形態による登録されたメタデータの判別及び抽出手順を示すフローチャートである。

【図５】

第２の実施形態によるメタデータの判別処理の詳細を説明するフローチャートである。

【図６】

メタデータとしてＸＭＬデータが登録されたディレクトリデータのデータ構成例を示す図である。

【図７】

本実施形態によるディレクトリ構造を説明する図である。

【図８】

本実施形態のファイル管理システムによるディレクトリデータの更新処理を説明するフローチャートである。

【図９】

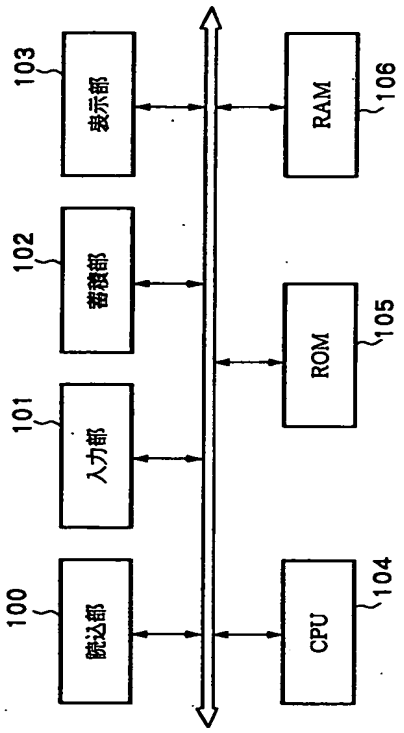
バイナリデータにメタデータを埋め込んだフォーマットの概観を示す図である。

【図１０】

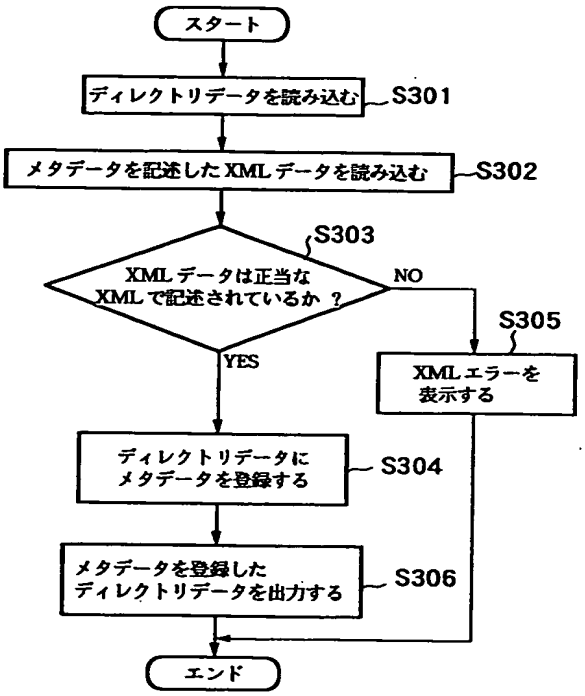
バイナリデータとメタデータをデータベースで管理する方法を概念的に示した図である。

【書類名】　図面

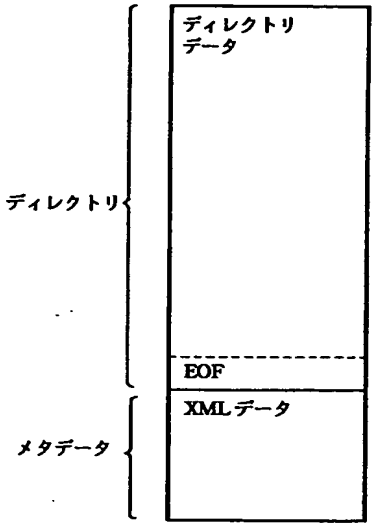
【図１】



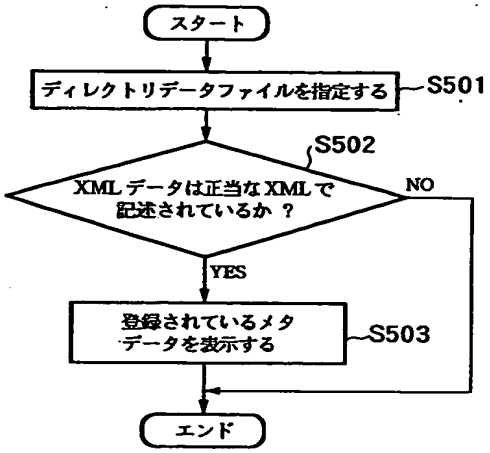
【図２】


スタート
ディレクトリデータを読み込む
S301
メタデータを記述したXMLデータを読み込む
S302
S303
XMLデータは正当な
XMLで記述されているか ?
NO
YES
S305
XMLエラーを
表示する
ディレクトリデータに
メタデータを登録する
S304
メタデータを登録した
ディレクトリデータを出力する
S306
エンド

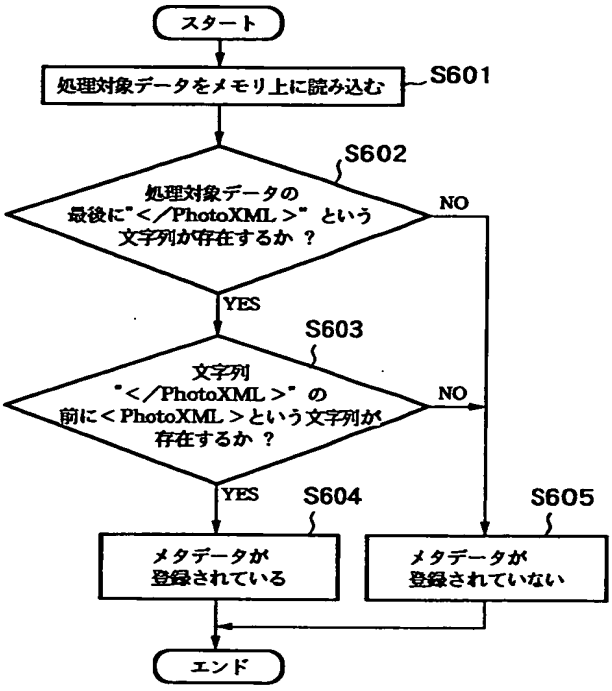
【図３】



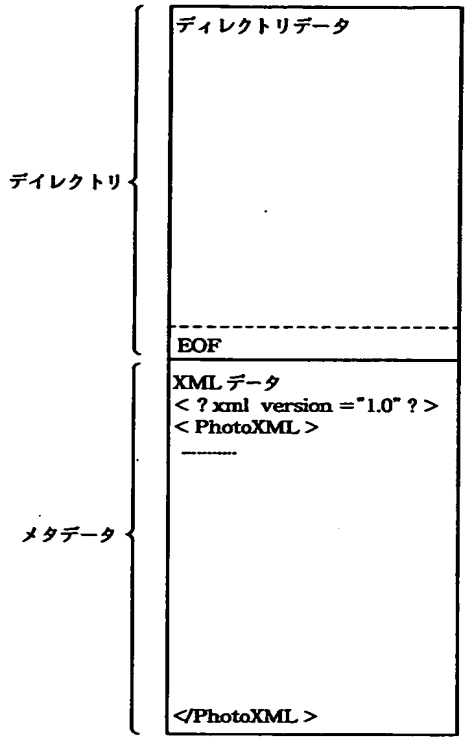
【図４】


スタート
ディレクトリデータファイルを指定する
S501
S502
XMLデータは正当なXMLで記述されているか？
NO
YES
登録されているメタデータを表示する
S503
エンド

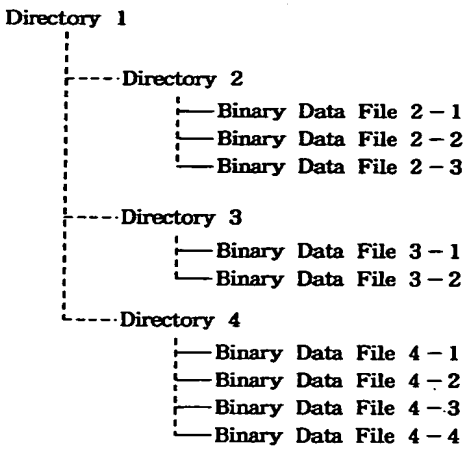
【図５】


スタート
処理対象データをメモリ上に読み込む
S601
S602
処理対象データの
最後に"＜／PhotoXML＞"という
文字列が存在するか ？
NO
YES
S603
文字列
"＜／PhotoXML＞"の
前に＜PhotoXML＞という文字列が
存在するか ？
NO
YES
S604
S605
メタデータが
登録されている
メタデータが
登録されていない
エンド

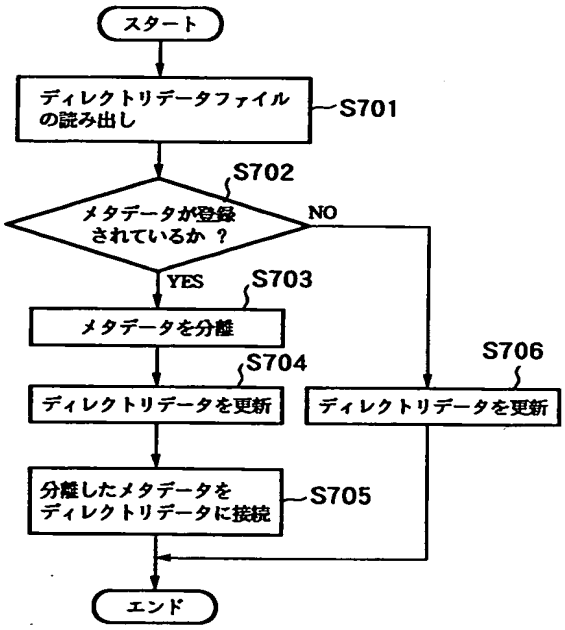
【図６】


ディレクトリデータ
ディレクトリ
EOF
XML データ
< ? xml version ="1.0" ? >
< PhotoXML >
メタデータ
</PhotoXML >

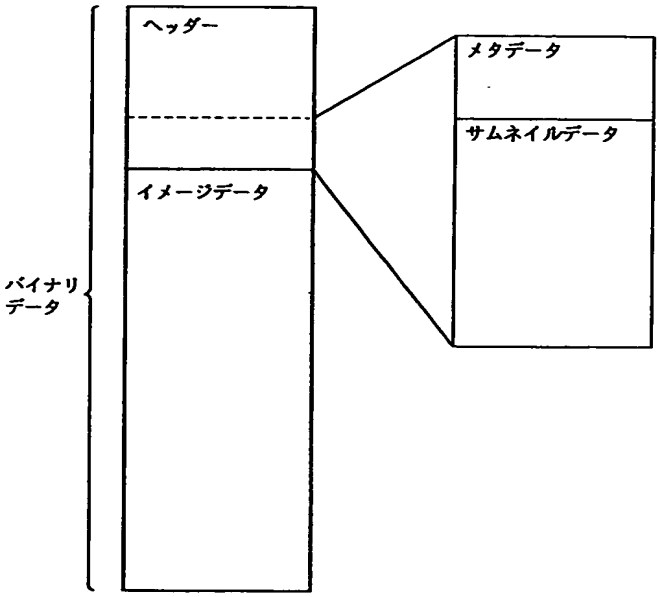
【図７】


Directory 1
Directory 2
Binary Data File 2－1
Binary Data File 2－2
Binary Data File 2－3
Directory 3
Binary Data File 3－1
Binary Data File 3－2
Directory 4
Binary Data File 4－1
Binary Data File 4－2
Binary Data File 4－3
Binary Data File 4－4

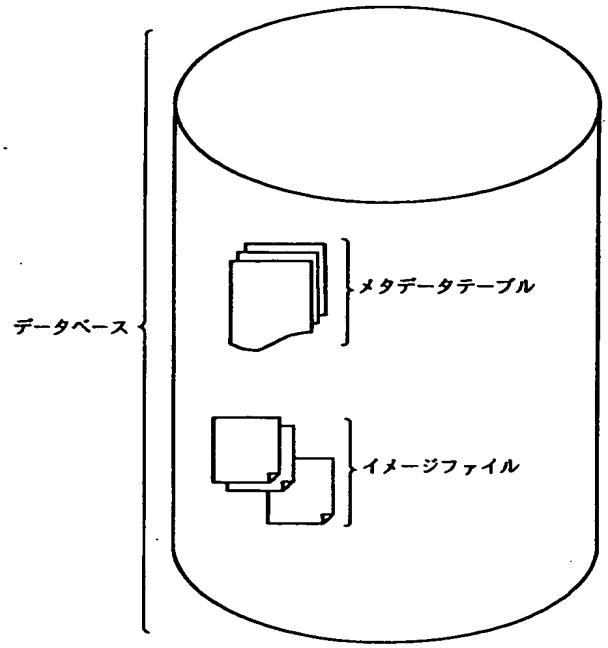
【図８】


スタート
ディレクトリデータファイルの読み出し
S701
S702
メタデータが登録されているか ？
NO
YES
S703
メタデータを分離
S704
ディレクトリデータを更新
S706
ディレクトリデータを更新
分離したメタデータをディレクトリデータに接続
S705
エンド

【図９】



【図１０】



【書類名】　要約書

【要約】

【課題】　メタデータをディレクトリデータの最後に接続することにより、ディレクトリで管理されている既存のデータに対し、既存のアプリケーションに影響を与えずに、ディレクトリ単位でメタデータを登録可能とする。

【解決手段】ステップＳ３０１において、ディレクトリ構造でファイルを管理するファイル管理システムによって各ディレクトリ毎に用意されているディレクトリデータを読み込む。ステップＳ３０２で、上記ディレクトリデータに付与すべきメタデータを読み込む。ステップＳ３０３において、上記読み込まれたメタデータが正当なＸＭＬで記述されていると判断された場合、ステップＳ３０４へ進み、上記ステップＳ３０１読み込まれたディレクトリデータの後に、ステップＳ３０２で読み込まれたメタデータを接続し、ステップＳ３０６において、得られたデータの全体をディレクトリデータファイルとして出力する。

【選択図】　図２

出　願　人　履　歴　情　報

識別番号　　〔０００００１００７〕

１．変更年月日　　１９９０年　８月３０日
［変更理由］　新規登録
住　所　東京都大田区下丸子３丁目３０番２号
氏　名　キヤノン株式会社